

A fuzzy knowledge base to support routine engineering design

John Dewey Jones*, Yao Hua

School of Engineering Science, Simon Fraser University, Burnaby, BC, Canada V5A 1S6

Received April 1995; revised November 1996

Abstract

Much of engineering design can be characterised as putting together variants of existing mechanisms to meet novel requirements. This paper describes an approach to engineering design in which fuzzy sets are used to represent the range of variants on existing mechanisms. Membership functions are chosen to reproduce the distinctions and classifications used by experienced designers. Methods are introduced to calculate the fuzzy performance range achievable by each component type, and a metric is suggested for the ranking of design candidates against design requirements. The underlying approach to design evaluation derives from that developed by Antonsson, Wood and Otto.

If the components model the design domain accurately, this architecture will always find a successful design, where one exists. However, finding this design may involve exhaustive search of the design space, and the time required for such an exhaustive search is typically far greater than is available for the task. The architecture is therefore augmented by the introduction of design agents, embodying heuristic rules to direct the search intelligently, so that a satisfactory design is found within a reasonable length of time.

As an example, the method is applied to preliminary design of a Stirling engine heat exchanger. © 1998 Elsevier Science B.V. All rights reserved.

Nomenclature

a	area (m^2)
B_n	Beale number
d	diameter of a heat exchanger tube (m)
E	Young's modulus (Pa)
f	friction factor
f_d	metric function, used to compare performance with requirements
f_D	metric function, used to combine different aspects of performance
h	film heat transfer coefficient ($\text{W}/\text{m}^2 \text{K}$)
k	fluid conductivity ($\text{W}/\text{m K}$)
l	length of a heat exchanger tube (m)

L	length of a heat exchanger (m)
n	number of tubes in heat exchanger
Q	heat flow rate per degree (W/K)
u	fluid velocity (m/s)
V	volume (m^3)
w_i	weighting factor
Δp	pressure drop across an exchanger (Pa)
ρ	gas density (kg/m^3)
μ, ν	membership functions

1. Introduction

We admire inventiveness in design, in part because of its rarity, in part because we believe it requires higher intellectual powers. Nevertheless, a rational

* Corresponding author. E-mail: jones@cs.sfu.ca.

designer will avoid novelty wherever possible. The time and effort required to detail, test and set up to manufacture a novel mechanism are orders of magnitude greater than required to modify an existing component. This is borne out by studies of practising engineers [2, 14], which show that designers reuse familiar solutions, and will not explore innovative ideas unless forced to do so by the repeated failure of attempts to adjust the old solution to the new problem. These observations underlie the approach developed in this paper: we provide the designer with a vocabulary of existing mechanisms, tools for modifying and combining these mechanisms, and a metric for comparing the resultant performance with the design requirements.

The system described here cannot guide the designer through all phases of the design process. In particular, it offers no support for geometric or spatial reasoning, which are critical parts of mechanical design. It operates at the verbal level of description, rather than in mathematical or graphic detail. But, as will be shown, even an inexact representation can yield rigorous, definite and useful conclusions. It can, for example, rule out large parts of the design space on the basis of a simple calculation.

2. Outline of the design process

We analyse routine design into the following stages:

- (i) Define the design requirements. Commonly, there will be some imprecision in these requirements.
- (ii) Assemble a list of candidate designs that might satisfy these requirements. This is a list of *names*, not of fully specified designs, each of which may cover a wide range of possible realisations.
- (iii) Calculate the performance ranges expected from each candidate, compare these with the design requirements, and rank the candidates on the basis of this comparison.
- (iv) For the leading candidate(s), go to the next level of refinement. We refine first by going from the general to the specific: having determined we need a diesel engine, we next ask if it should be a direct or indirect-injection diesel. When we arrive at the specific, we next refine by going from the whole to the part, e.g., from design of the engine to design of the piston.

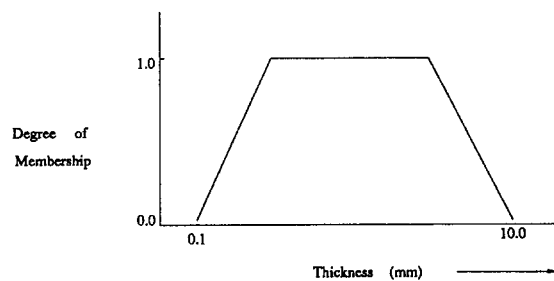


Fig. 1. Membership function for “thickness” parameter of “rubber band”.

(v) We rank the candidates according to their predicted performance at each level of refinement until we reach the lowest level of the classification and component hierarchies: mechanisms without components, characterizable by a small number of parameters whose values lie within short intervals.

A class of mechanisms may appear promising at the first stage, but all of its subclasses may fall short of the requirements. Therefore, it is essential that the design procedure be capable of *backtracking*, that is, going back to develop a second design candidate if the initial choice performs poorly.

2.1. Building a knowledge base of components

To develop a fuzzy knowledge base to support conceptual design, we create and maintain a library of components. A component is characterised by a number of parameters, and by *its* components, if any. The feasible range of values for a parameter is represented by a fuzzy set; in some cases this set will be a fuzzy number, in other cases it may represent a choice over a discrete number of alternatives.

For example, a rubber band might be characterised by the parameters *length*, *width*, *thickness* and *material*. *Material* is represented by a discrete set, for example, {*hard rubber*, *soft rubber*}, and the properties of each type of material are in turn represented by fuzzy numbers. The membership functions defined for these parameters represent the imprecision involved in the expression “rubber band”. It is clear that there is no precise upper bound to the thickness of a rubber band, yet it is also clear that a rubber cube having side length 1 m is not a rubber band. We might represent the parameter “thickness” by a membership function like that shown in Fig. 1.

In representing the set of choices for the material, there is a trade-off between the level of detail used in specifying the discrete choices and the precision with which the properties of each alternative can be specified. We could, for example, offer the single choice “rubber”; its properties would then span the entire range of values found in any rubber. Alternatively, we could specify a very detailed list of choices, perhaps taken from a manufacturer’s catalog. The properties of each choice would still be represented by fuzzy numbers, reflecting the uncertainty in the tests used to measure them and the unavoidable variations in the rubber’s composition and processing, but these numbers would be much more sharply peaked than those for “rubber”.

This illustrates one way in which the fuzzy knowledge base is built up: we can construct representations of the properties of imprecisely defined sets from the tabulated properties of the members of the set. Formally, suppose “rubber” is taken to include the manufacturer’s products $R_1, \dots, R_n$.

For the i th sample, let (E_i, μ_i) be the fuzzy number representing Young’s modulus, a physical property corresponding to elasticity. Then the Young’s modulus of “rubber” can be defined as the fuzzy number (E, μ) , where

$$E = \bigcup_i E_i \quad \text{and} \quad \mu = \bigwedge_i (\mu_i). \quad (1)$$

Unfortunately E , so defined, may not be convex. For example, the range of products may contain types of rubber with Young’s moduli of 1.4 and 2 GPa, but none with a modulus of 1.6 GPa. If such gaps are not too large, we deal with this by redefining the membership function of E as the convex hull of μ . This may lead to costly backtracking during the design process. In some cases, ensuring convexity is worth this penalty; where it is not, we treat the different types of rubber as separate materials, and do not attempt to create an inclusive superclass.

The second way in which the knowledge base is built up is through component modelling: guided by an expert in the field, we put imprecise bounds on the parameters of a component, then deduce the corresponding limits on its performance.

The performance of a component is a function of its parameters and of its boundary conditions. The

distinction between a component parameter and a boundary condition is sometimes difficult to make; roughly speaking, a parameter is “internal” to the component, while the boundary conditions are imposed by the context in which the component is used. The boundary conditions for the design of a particular component may be given explicitly or implicitly in the design requirements, or they may follow from choices made in the design of neighbouring components. Given the design parameters, boundary conditions, and governing equations of a component, methods exist [19] for calculating the fuzzy number(s) representing the component’s performance.

A shortcut is sometimes available here. Rather than using the most detailed equations to calculate component performance, we can use approximate expressions whose degree of approximation is less than or comparable with the level of imprecision in the parameters. For example, Stirling engine designers often make use of the Beale number [15], which establishes the empirical relationship

$$\frac{\text{Power} \cdot \text{Output}}{\text{Swept} \cdot \text{Volume} \times \text{Engine} \cdot \text{Speed} \times \text{Mean} \cdot \text{Pressure}} = B_n, \quad (2)$$

where B_n is a fuzzy number corresponding to “about 0.15”. The Beale number is one example of a rule of thumb used in design, and illustrates that the present approach can capture informal knowledge as well as exact reasoning.

2.2. Using the knowledge base for design

When the knowledge base is to be used for design, the user will supply a number of *design requirements*. These will also be most naturally represented as fuzzy numbers. An engine designer, for example, is more likely to be asked for “a sporty four-cylinder engine” than for an engine with a power output of 500.00 kW. There will typically be a number of requirements on a design. Some may be expressed as fuzzy targets for performance, others as fuzzy upper or lower bounds on variables – e.g., “light”, “cheap”, or “quiet”.

Note that there is a difference in interpretation between the membership functions used for the design parameters and those used for the performance requirements. In the former case, the degree of membership reflects our estimate of how well a given value falls within the meaning of an imprecisely defined

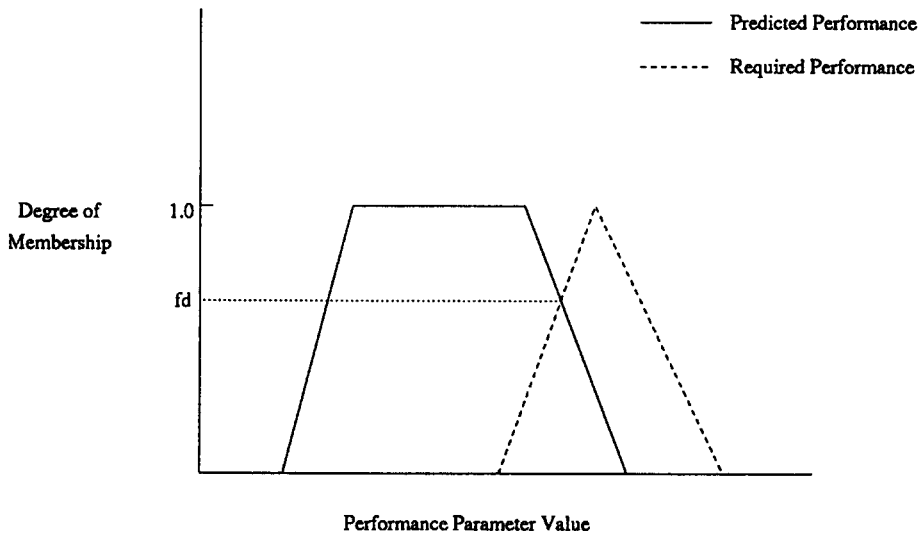


Fig. 2. Comparing predicted and required performance.

word in the vocabulary; in the latter, the degree of membership represents the *desirability* of a performance parameter achieving a given value.

We now require a metric, which we will call a *match*, for comparing the fuzzy numbers representing the predicted and desired performance for a given component or components. Let the performance variable under discussion have predicted value (X, μ) and let the desired performance for the same variable be (Y, ν) . Then the match between prediction and requirement is a real number in the range $[0, 1]$ given by $f_d(X, Y, \mu, \nu)$. This metric should have the following characteristics:

- (i) Given a second component with performance (X, μ') and $\mu'(x) \geq \mu(x) \forall x \in X \cap Y$, $f_d(X, Y, \mu', \nu) \geq f_d(X, Y, \mu, \nu)$.
- (ii) Given a second requirement with preference (Y, ν') and $\nu'(x) \geq \nu(x) \forall x \in X \cap Y$, $f_d(X, Y, \mu, \nu') \geq f_d(X, Y, \mu, \nu)$.
- (iii) If $X \cap Y = \emptyset$, $f_d(X, Y, \mu, \nu) = 0$.

The following definition of f_d satisfies these conditions, though other definitions are possible:

$$f_d = \max_{x \in X \cap Y} (\min(\mu(x), \nu(x))). \quad (3)$$

This is illustrated in Fig. 2.

In general, we will have a number of functional requirements. How do we combine the matches for each requirement to get an overall measure of suitability?

Let us denote this combined metric by $f_D(d_1, \dots, d_n)$, where d_i is the value of f_d for the i th performance requirement. The operation f_D should have the following characteristics:

- (i) $f_D(d_1, \dots, d_n) = 0$ if $\exists i: d_i = 0$ ($1 \leq i \leq n$),
- (ii) $f_D(d_1, \dots, d_n) = d$ if $d_1 = \dots = d_n = d$,
- (iii) $f_D(d_1, \dots, d_i, \dots, d_n) \leq f_D(d_1, \dots, d'_i, \dots, d_n)$ iff $d_i \leq d'_i$.

One candidate operation is

$$f_D(d_1, \dots, d_n) = d_1^{w_1} \times \dots \times d_n^{w_n}, \quad \sum_{i=1}^n w_i = 1, \quad (4)$$

where w_i represent weights which can be attached to the different requirements. Another possibility is

$$f_D(d_1, \dots, d_n) = \min_{1 \leq i \leq n} (d_i). \quad (5)$$

The choice of f_D corresponds to the choice of a particular design strategy. Otto and Antonsson examine this choice in [12]. Their use of fuzzy sets in design differs from the approach described here, but their discussion can usefully be applied to the present case. The earlier work of Diaz [5] is also relevant.

2.3. Refining the design

In the first stage of the design process, the designer faces a choice between a number of different mechanisms to meet the given requirements. For example,

we might want to compare Stirling, diesel and gasoline engines as possible prime movers for city buses. An initial selection may be made by using the functions f_d and f_D to compare the imprecise performance for each alternative with the given requirements.

This early stage of design may serve to rule out some candidate solutions, but the imprecise performance parameters of most candidates will have broad plateaux. This is what we would expect, reflecting the fact that many alternatives might be pursued to meet the requirements. To proceed with the design, it is necessary to refine the representation, distinguishing the subclasses of each mechanism recognised in the technical vocabulary of the field.

In some cases the levels of detail afforded by the technical vocabulary may go all the way down to full specificity, for example, to component descriptions from an engineering catalog. In other cases – if, for example, many of the components of a proposed design are to be constructed *de novo* – we must augment the distinctions already present in the vocabulary. For example, we may distinguish between “large”, “medium” and “small” variants of a component, associating each variant with an appropriate set of membership functions for its design parameters. These distinctions are intuitive and simple to implement. Further distinctions can then be made by the use of linguistic “hedges” [20] to distinguish, for example, “small” from “very small”.

3. An implementation

We have built an interactive design system using a constraint-based, object-oriented logic programming language called “Echidna” [9]. The knowledge bases for this system have a two-part structure: there is a set of objects, representing possible design *components*, and a second set of objects – “agents” – expressing design *strategies*.

We represent the components within a classification hierarchy: a regenerative heat exchanger, for example, is a subclass of the more general class of heat exchangers, and inherits some of its properties from its superclass. Each component is characterised by a set of parameters, representing its physical attributes, and a set of constraints on those parameters, expressing the laws of physics governing its behaviour.

Design agents are written in the same language, but make greater use of choice between clauses. This difference reflects the fact that strategies are usually heuristic: if one line of development fails, we go back and try another.

This separation between components and agents increases the re-useability of the knowledge recorded in the knowledge bases. The agents employ heuristics obtained from experts in the chosen context, a process grandiloquently referred to in the expert systems literature as “knowledge engineering”. We believe the knowledge engineer’s task is greatly facilitated by using fuzzy sets to represent component types corresponding to the vocabulary naturally used by the domain expert.

3.1. Compilation of the knowledge base

To speed the design process, we would like the performance of each component to be “pre-compiled” – calculated in advance of the design session from the imprecise values of the design parameters. For some aspects of performance this may be possible – for example, imprecise limits on the mass and volume of a component can usually be calculated directly from the design parameters. But where the performance depends strongly on the boundary conditions, we must generate the performance parameter values during the design session, as the boundary conditions are supplied.

We have developed an algorithm for calculating the performance of a component from the fuzzy sets defining its design parameters in [19], based on interval arithmetic applied to α -cuts of the sets. Efficient use of this algorithm requires use of monotonicity analysis, as described in [11].

4. Example

In this section, we develop an example for the case of heat exchanger design in a Stirling engine application. For readers unfamiliar with this area, it may be helpful to provide a brief background.

A heat exchanger is a device for the transfer of heat between a hot and a cold fluid. One common design is the shell-and-tube exchanger, in which one fluid flows through a set of tubes and the other flows over the outside of the tubes through an enclosing shell.

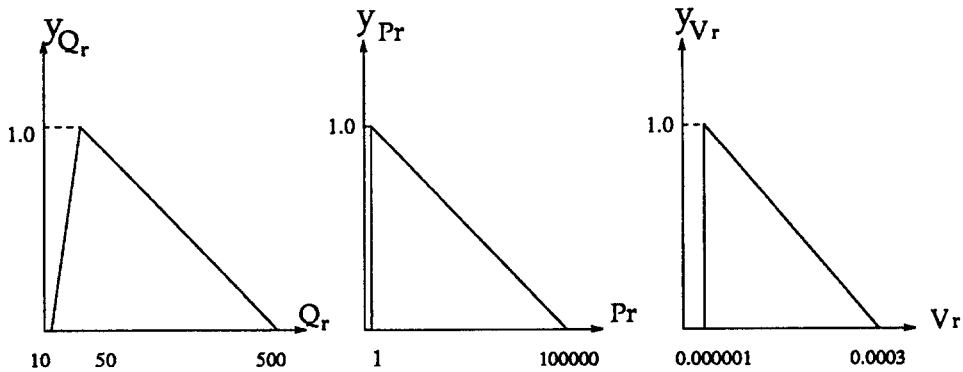


Fig. 3. Requirements for design.

Other designs sometimes used in Stirling engines are the finned exchanger, in which one fluid flows in and out of a closed volume, the other fluid surrounding the finned outer surface of the volume; and the plain exchanger, in which the fins are omitted.

In the particular example which we will consider, there are following specific requirements for the design:

1. The heat flow per unit degree temperature difference through the heat exchanger walls should be about 50 W per degree; lower than 10 W per degree is definitely not good enough, and greater than 500 W per degree would be overdesigned.

2. The pressure drop across the exchanger should be as low as possible, and certainly less than 100 kPa.

3. The internal volume of the heat exchanger should be as small as possible, and certainly less than 300 cm³.

These requirements are represented as fuzzy numbers in Fig. 3.

At the first stage of the design process, the designer eliminates all but one of the types of exchanger. Next, an approximate size is selected for the chosen type.

4.1. First stage

We begin with the component types stored in the knowledge base and fuzzy numbers representing the design requirements and boundary conditions. We copy the selected component types from the knowledge base and add the boundary conditions, thus obtaining fuzzy sets representing component performance.

4.1.1. Performance parameter expressions (PPEs)

The performance parameter values are calculated using equations stored in the knowledge base, as follows. We denote the heat flow per unit degree temperature by Q , the pressure drop across the exchanger by P , and the internal volume of the exchanger by V .

In equation form, the PPEs for the shell-and-tube heat exchanger can be expressed as

$$Q_{\text{tube}} = ha, \quad (6)$$

$$P_{\text{tube}} = \frac{f\rho u^2 l}{2d}, \quad (7)$$

$$V_{\text{tube}} = \frac{\pi n d^2 l}{4}, \quad (8)$$

where a is the surface area of the tubes, f the friction factor, ρ the gas density, u the mass flow rate, l the tube length, d the tube diameter, and n the number of tubes. In order to describe this problem completely, further expressions are needed, relating the variables on the right-hand sides of these equations to other quantities such as Reynolds number. Whatever intermediate quantities are introduced, the complete set of equations will specify the performance parameters in terms of the design parameters and boundary conditions only.

Similar expressions can be written down for the performance of the finned and plain heat exchangers.

4.1.2. Design parameters

The design parameters are represented by trapezoidal or triangular fuzzy numbers. In the shell-and-tube heat exchanger these parameters are the tube

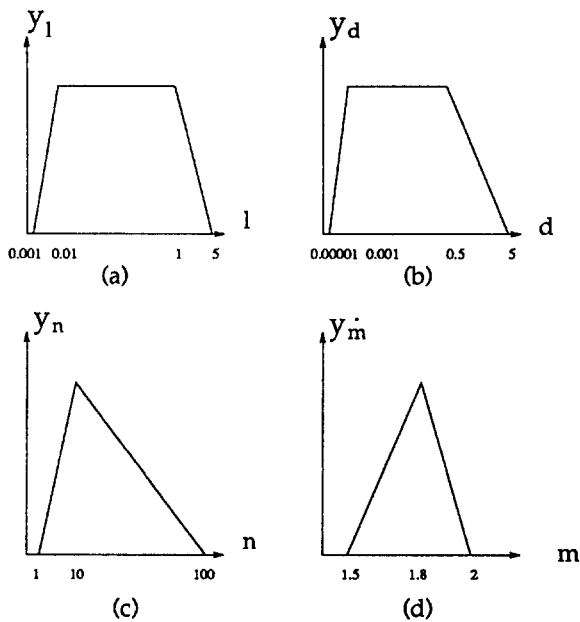


Fig. 4. Design parameters for a shell-and-tube heat exchanger.

length l , the tube diameter d , and the number of tubes n . The mass flow rate of water $\dot{m}$ is a boundary condition.

Fig. 4 shows membership functions for each design parameter for the shell-and-tube heat exchanger. The plain heat exchanger has two design parameters, the cylinder diameter d , and the cylinder height l shown in Figs. 5(a) and (b). The finned heat exchanger is characterised by four parameters: number-of-fins n shown in Fig. 5(c); cylinder diameter d , cylinder height l , and fin height h shown in Figs. 5(a), (b) and (d), respectively.

4.1.3. Performance parameters (PPs)

Given the input parameters, performance parameters can be determined by the PPEs and a pole computation algorithm [19]. Their values are shown in Fig. 6. Since they are very broad ranges, a logarithmic scale is used for drawing their graphs. The heat flow rate per degree temperature difference of the exchanger is shown in Fig. 6(a). The performance value (Q_{tube}, μ) is drawn as a solid line and the corresponding requirement (Q_r, v) as a dotted line. The design match f_d for this requirement is unity: the required value is well within the predicted performance limits. Similarly, the

design matches for pressure drop and internal volume are unity, giving an overall performance measure of unity.

For the plain heat exchanger, since there is no intersection between the heat flow rate per unit degree temperature difference and the corresponding requirement (Fig. 7(a)), the match for heat flow rate per unit degree temperature is zero. This leads to a zero overall measure.

For the finned exchanger, the match for heat flow is close to 0.5 (Fig. 7(c)), and so is the overall match. Figs. 7(b) and (d) show the requirements and performance for internal volume for the plain and finned heat exchanger, respectively.

Using the metrics f_d and f_D to compare the imprecise performance for each alternative with the given requirements, the shell-and-tube heat exchanger is chosen. The next stage allows us to refine this result.

4.2. Refining stage

The shell-and-tube heat exchanger is selected for the continuing design. Since there are wide plateaux for each parameter, it is necessary to refine each representation. The knowledge base supporting heat exchanger design contains fuzzy models of components for different sizes of the shell-and-tube heat exchanger, for example, “VerySmall”, “Small”, and so forth. Each size of exchanger has its own diameters and lengths, represented by trapezoidal fuzzy numbers shown in Fig. 8.

As before, results can be obtained for the performance parameters. These are shown in Fig. 9. From left to right curves are drawn for the “VerySmall”, “Small”, “Medium”, “Big” and “VeryBig” shell-and-tube heat exchangers, respectively. Dotted lines are plotted for the functional requirements.

Looking at the first diagram, we can read off the values of f_d for each size variant by noting the height at which the dotted line cuts each of the solid lines. We do the same thing for the remaining two diagrams; then, for each size variant, we take f_D to be the minimum of the three values of f_d just obtained. Specifically,

$$f_{D\text{VeryBig}} = 0, \quad (9)$$

$$f_{D\text{Big}} = 0.2, \quad (10)$$

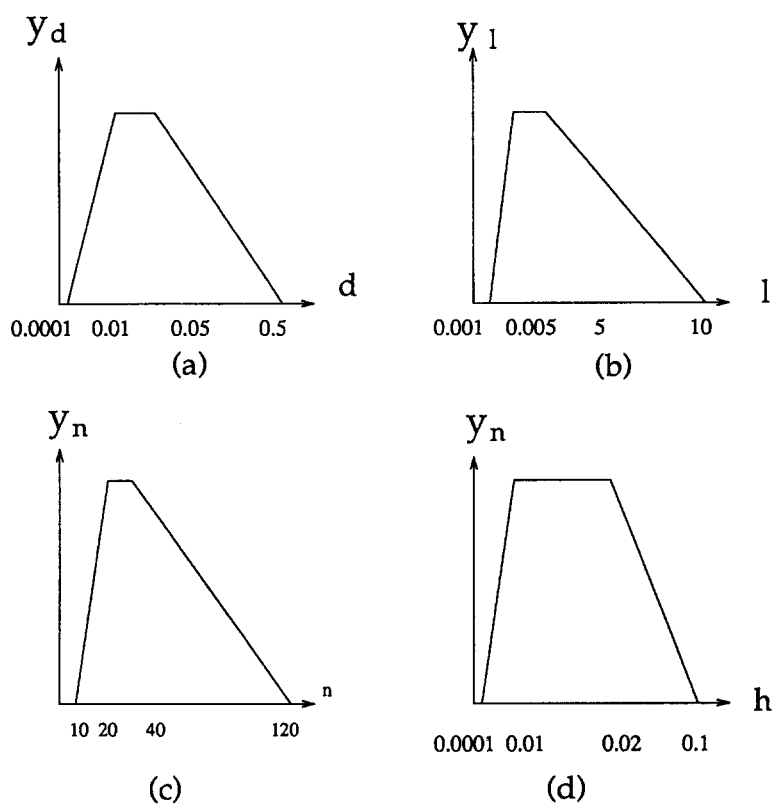


Fig. 5. Design parameters for plain and finned heat exchangers.

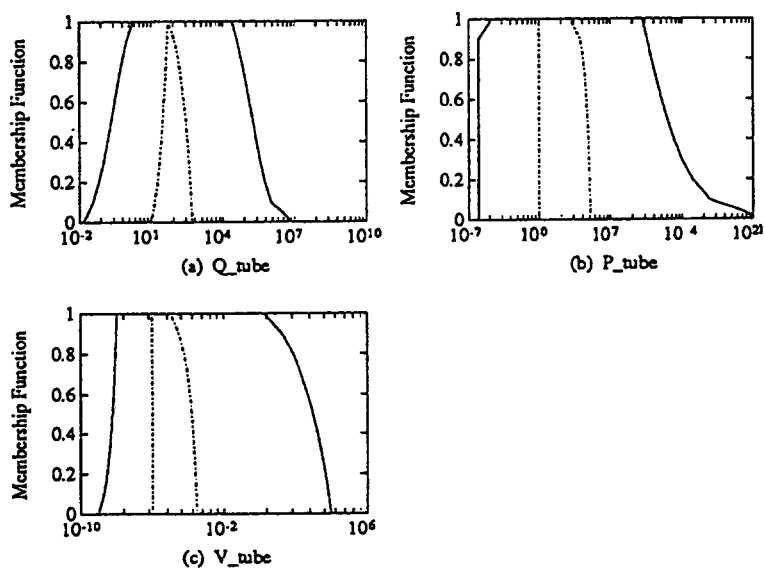


Fig. 6. PPs and FRs for a shell-and-tube heat exchanger.

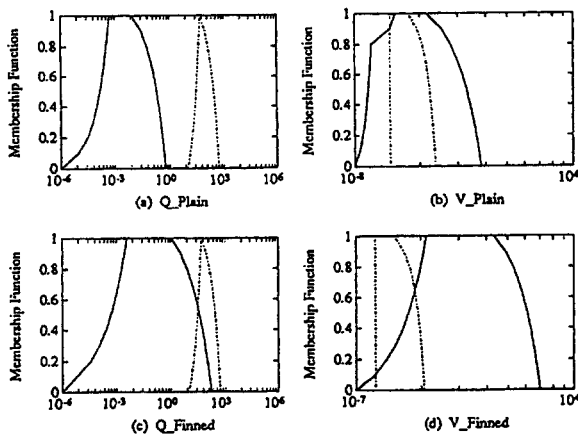


Fig. 7. PPs and FRs for plain and finned heat exchangers.

$$f_{D\text{Medium}} = 0.3, \quad (11)$$

$$f_{D\text{Small}} = 0.8, \quad (12)$$

$$f_{D\text{VerySmall}} = 0.2. \quad (13)$$

The maximum measure is 0.8 for the “Small” size of the shell-and-tube heat exchanger, which is therefore selected for further refinement.

4.3. Precise stage

After the first and second stages, design can be focused on a narrow range, in this case the “Small” shell-and-tube heat exchanger. The methods introduced in this paper can take us no further. However, we are now ideally placed to apply Antonsson and Wood’s “backward path” method [17] to obtain precise values. The details of this method will not be discussed here; however, for completeness we note that application of this method to our partially specified design leads to a fully specified design, with design parameter values $d = 0.01$ m, $l = 0.01$ m, $n = 5$ and $\dot{m} = 1.5$ kg/s, yielding performance parameter values $Q = 34$ W/°C, $\Delta P = 7826$ Pa and $V = 4$ cm³.

4.4. Related work and discussion

Knowledge bases have been written which can perform routine design tasks efficiently; for example, Brown and Chandrasekaran’s “AIR-CYL” system [3]. But the knowledge used to support this design is expressed in very specific terms, and could not

be reused to support design of a different class of components. This limits the applications to which this approach may be applied: the development of a knowledge base is expensive, and there are relatively few applications for which the cost of developing an original knowledge base could be justified.

In contrast, we have efforts such as the *How Things Work* project at Stanford [4], which seek to deduce engineering decisions from the most general basis, namely, the laws of physics. A knowledge base written in terms of the laws of physics would be eminently reusable, but to devise strategies that could design a can-opener, starting from the laws of mechanics, is a daunting task.

One response to the apparent intractability of this task has been to seek a simplified form of physics, closer to that used in common-sense reasoning. Forbus [8] and DeKleer [6] have developed versions of a “qualitative physics”, and several publications have described attempts to use this physics to support design reasoning, e.g., [1, 13].

The “VEXPERT” system for V-belt design developed by Dixon [7] makes use of utility curves to evaluate a design’s performance against several criteria; these curves are analogous to the preference functions we associate with performance parameters, though Dixon combines the utilities using the formula

$$f_D = \sum_i w_i d_i \quad (14)$$

which does not conform to the conditions suggested here.

The work of Professor Antonsson and his students, notably Dr. Wood and Dr. Otto, has strongly influenced the present research. This work has been quite widely reported [17, 18, 12]. As in the present work, Antonsson and Wood’s approach makes use of the fuzzy calculus developed by Zadeh. Their approach is applied at a stage in the design process when the design parameters are known. Preference functions are associated with sets of possible values for each of the design parameters; unlike the present approach, however, these functions denote the designer’s preference for using a particular value of the design parameter. From the equations describing the physics of the component, referred to as the “Performance Parameter Expressions”, or “PPEs”, it is possible to calculate a fuzzy number representing

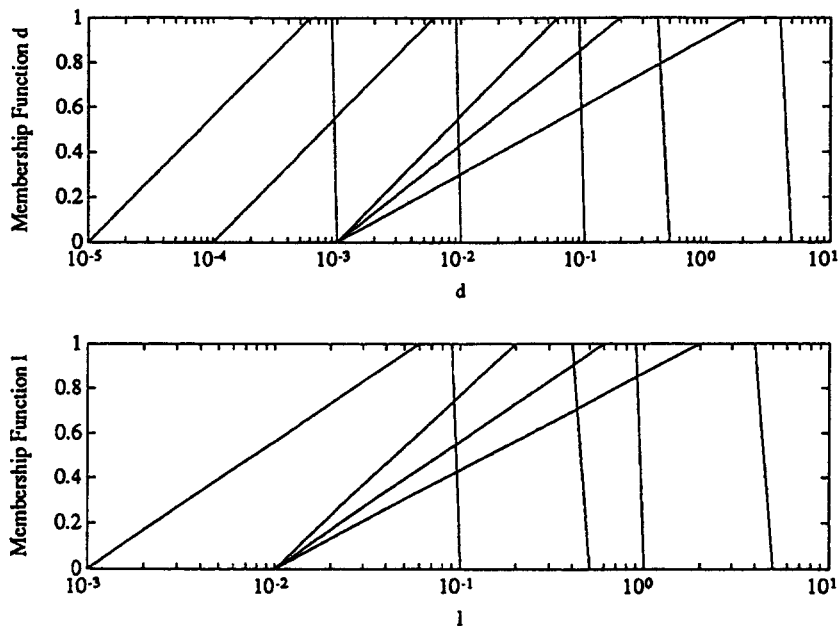


Fig. 8. Design parameters for a shell-and-tube heat exchanger.

the performance of the design. This fuzzy number associates with each numerical value of the performance parameter a value of the membership function for that parameter. In this case, however, the membership function does not denote the designer's preference that the performance take that value; rather, it denotes how desirable that value would be if the designer were concerned only with achieving the targetted values for the design parameters. The peak preference of a performance parameter will not in general coincide with the required value for that parameter, and one or more of the design parameters must therefore be adjusted from its most preferred value for the performance to meet the requirements. The method offers useful guidance in making this adjustment, in the form of the "backwards path". This path depends on the observation that to achieve a performance requirement having degree of membership μ_1 in the calculated fuzzy set for that performance parameter, at least one of the relevant design parameters must shift to a value having preference μ_1 or lower. Further assistance in identifying the parameter(s) to be changed comes from the "gamma-level measure", an easily calculated measure of the sensitivity of the performance parameter

to a change in preference of each of the design parameters.

The approach described in the present paper differs from that of Antonsson and his students in several ways. Firstly, we associate membership functions with design parameters to reproduce the vocabulary of a domain expert, rather than to express a preference for using particular values. It appears difficult to us to attach rational preferences to the values of isolated design parameters; for example, given a choice among a range of steel alloys, should we prefer the cheaper alloys or the stronger ones? We can see no sensible way of answering this question before seeing the impact the choice of material has on the final design.

By storing membership functions in a persistent knowledge base, we make the knowledge engineer responsible for the choice of an appropriate membership function. Antonsson's approach leaves this responsibility to the designer, but the typical designer is unlikely to be familiar with fuzzy set theory, and therefore may not find membership functions a natural way of expressing his expertise.

Secondly, we propose creating and maintaining a library of component types. This allows much of the computation involved in predicting the expected

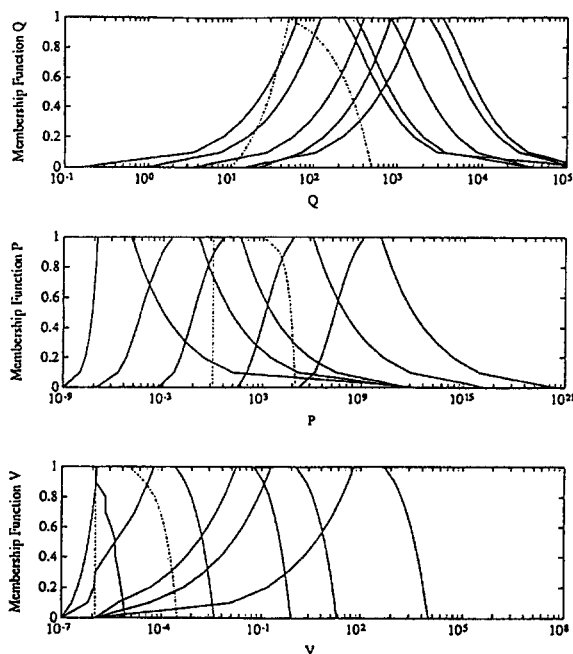


Fig. 9. PPs and FRs for a shell-and-tube heat exchanger.

performance of a component to be done when the knowledge base is compiled, rather than in the course of the design session. Moreover, the performance of many alternatives can be computed in parallel.

4.4.1. Neurofuzzy methods

There has been growing interest recently in “neurofuzzy” methods [10]. It is natural to ask whether this approach would provide an alternative to the approach described here.

One neurofuzzy approach to the design problem would be to replace our library of components with a library of neural nets. For each net we have a set of “input” nodes, representing our design parameters, and a set of “output” nodes, representing the performance parameters. The connectivity of the net is set so that the design parameters fix the performance parameters via the appropriate set of performance parameter expressions, or PPEs. Now, to solve a particular design problem, we peg the net outputs at the desired levels, and use back-propagation to determine the corresponding levels for the input parameters. How would such an approach compare with that outlined in the body of this paper?

This neurofuzzy approach appears to offer an alternative for the *final* stage of the design process, the stage at which we instead used the method developed by Antonsson and Wood. For the earlier stages of the process, which are those of chief concern in the present paper, there are three obstacles to the use of the neurofuzzy approach. First, in contrast to the applications in which neural nets have been most successful, we start here from an explicit and accurate model of device behaviour, in the form of the performance parameter expressions. To train nets to reproduce the functionality of these expressions would be time-consuming and perhaps redundant.

Second, it is not clear how the neurofuzzy approach could select the component types to be investigated. Our representation uses interval arithmetic to pre-calculate the range of performance that may be expected from each component type, and this pre-calculated range provides a basis for initial component selection. The neural net representation, by contrast, would predict for each type the particular performance corresponding to the default settings of the input parameters, which would not provide a suitable basis for selection.

Third, the approach developed in the body of this paper supports mixed-initiative design; that is, the designer and the computer alternately make choices that reduce the size of the design space. The intervention of the designer, and of design heuristics, is necessary because of the immense size of the design space for problems of practical interest. It is for this reason, also, that we have found it necessary to incorporate backtracking in our method. In investigating the neurofuzzy alternative, it would be essential to study how rapidly solution time increased with the number of design parameters when the problem is solved by pure back-propagation, unaided by human insight, heuristics or intelligent backtracking. Note that real-world design problems, unlike the example presented here, typically involve several hundred design parameters.

Acknowledgements

The authors would like to acknowledge the support of the National Science and Engineering Research Council of Canada and the Science Council of British

Columbia. We would like to thank our colleagues in the Logic and Functional Programming Group and the Expert Systems Laboratory at Simon Fraser, in particular Bill Havens. Thanks are also due to Professor Antonsson, Dr. Kris Wood and Dr. Kevin Otto, who patiently corrected our understanding of their work in several email exchanges.

References

- [1] H.G. Barrow, VERIFY: a program for proving correctness of digital hardware designs, in: D. Bobrow, Ed., *Qualitative Reasoning about Physical Systems* (MIT Press, Cambridge, MA, 1985).
- [2] G. Basalla, *The Evolution of Technology* (Cambridge Univ. Press, Cambridge, UK, 1988) 21, 26, 208–209.
- [3] D.C. Brown and B. Chandrasekaran, *Design Problem Solving* (Morgan Kaufmann, Los Altos, CA, 1989).
- [4] M.R. Cutkowsky and J.M. Tenenbaum, Research in computational design at Stanford, *Res. Eng. Des.* **2** (1990) 53–59.
- [5] A.R. Diaz, Fuzzy-set-based models in design optimization, in: S.S. Rao, Ed., *Advances in Design Automation – 1988*, Vol. DE-14 (ASME, New York, 1988) 477–485.
- [6] J. deKleer and J.S. Brown, A qualitative physics based on confluences, in: J.R. Hobbs and R.C. Moore, Eds., *Formal Theories of the Commonsense World* (Ablex, 1988).
- [7] J.R. Dixon and M.K. Simmons, Expert systems for design: standard V-belt drive design as an example of the design-evaluate-redesign architecture, *Proc. ASME Computers in Engineering Conf.*, Boston, MA, August (1984).
- [8] K.D. Forbus, Qualitative process theory, *Artificial Intelligence* **24** (1984) 85–168.
- [9] W.S. Havens, Echidna constraint reasoning system: programming specifications, *Proc. Computational Intelligence*, Milano, Italy, September (1990).
- [10] B. Kosko, *Neural Networks and Fuzzy Systems* (Prentice-Hall, Englewood Cliffs, NJ, 1991).
- [11] Dong, Li and J. Jones, A sense of scale, *IEEE Pacific Rim Conf. Communications, Computers and Signal Propagation*, May (1993).
- [12] K.N. Otto and E.K. Antonsson, Trade-off strategies in engineering design, *Res. Eng. Des.* **3** (1991) 87–103.
- [13] T. Tomiyama, T. Kiriya, H. Takeda, D. Xue and H. Yoshikawa, Metamodel: a key to intelligent CAD systems, *Res. Eng. Des.* **1** (1989).
- [14] D.G. Ullman, L.A. Stauffer and T.G. Diettrich, Preliminary results of an experimental study of the mechanical engineering design process, in: M.B. Waldron, Ed., *Proc. NSF Workshop on the Design Process*, Oakland, CA, 8–10 February (1987) 145–188.
- [15] G. Walker, Elementary design guidelines for stirling engines, *Proc. 14th IECEC* (American Chemical Society, Boston, 1979).
- [16] G. Walker, *Stirling Cycle Machines* (Oxford University Press, Oxford, 1980).
- [17] K.L. Wood and E.K. Antonsson, Computations with imprecise parameters in engineering design: background and theory, *ASME J. Mechanisms, Transmissions and Automation in Design*, **111** December (1989) 616–625.
- [18] K.L. Wood and E.K. Antonsson, Computations with imprecise parameters in engineering design: applications and examples, *ASME J. Mechanisms, Transmissions and Automation in Design* **111** December (1989) 616–625.
- [19] H.Q. Yang, H. Yao and J. Jones, Calculating functions of fuzzy numbers, *Fuzzy Sets and Systems* **55**, 10 May (1993) 273–283.
- [20] L. Zadeh, The concept of a linguistic variable and its application to approximate reasoning, *Inform. Sci.* **8** (1975) 199–249.